REMARKS/ARGUMENTS

Claims 1-15 and 17-62 are pending in the application. Claims 25-36 and 55-61 are withdrawn, claims 1, 10, 13, 19, 21-23, 37, 45, and 51-53 are amended, and claim 62 is new. As discussed below, all of the claims are in condition for allowance. **But if after considering this response the Examiner does not allow all of the claims, then the Applicants' attorney requests that the Examiner contact him to schedule and conduct a telephone interview before issuing a subsequent Office Action.**

The Phone Interview With The Examiner On September 27, 2010

The Applicants' attorney thanks the Examiner for participating in a telephone interview with the Applicants' attorney, Mr. Bryan Santarelli, and coinventors Mr. John Rapp and Mr. Scott Hellenbach, on September 27, 2010.

Non-Compliant Information Disclosure Statement (Section 3 of the Office Action)

The Applicants' attorney submits with this Response to Office Action an IDS form that corrects the deficiencies pointed out by the examiner, and, therefore, the Applicants' attorney requests that the examiner withdraw this objection.

Objection To The Specification (Section 4 of the Office Action)

The Applicants' attorney has amended the title, and requests the examiner to approve the amended title and withdraw this objection.

Objections To The Claims (Sections 5-13 of the Office Action)

The Applicant's attorney has amended claims 10, 13, 19, 21, 22, 23, 37, 45, and 53 as requested by the examiner (except where such amendments are moot in view of other amendments to these claims), and, therefore, the Applicants' attorney request that the examiner withdraw these objections.

<u>Rejection Of Claims 10-15, 17-18, And 37-50 Under 35 U.S.C. § 112, First Paragraph, As Failing To Comply With The Enablement Requirement (Sections 14-17 of the Office Action)</u>

As Mr. Rapp and Mr. Hellenbach explained to the examiner during the September 27 phone interview, it was known in the art at the time that the application was filed how to "generate data under the control of [an] application such that the generated data includes only non-data-destination information" as recited, *e.g.,* in claim 10. For example, in an embodiment, a software application may include with the data a data type, and a data-transfer object assigned to the data type may be structured to transfer the data to a particular location based on the data's type. Because the data-transfer object may be constructed separately from, and may be modified more easily than, the application software, this technique may allow one to change the destination of the data without engaging in the often lengthy process of modifying, recompiling, and retesting the application software. During the interview, the examiner stated that he understood this explanation, and, therefore, that he would withdraw this rejection. Therefore, the Applicants' attorney requests that the examiner withdraw this rejection.

<u>Rejection Of Claims 10-12, 14-15, 17-18, 37, 39-43, 45, and 47-49 Under 35 U.S.C. § 102(b) As Being Anticipated By U.S. Patent 4,703,475 To Dretzka (Sections 18-41 of the Office Action)</u>

For the reasons below, the Applicants' attorney requests that the examiner withdraw these rejections.

<u>Claim 10</u>

Claim 10 as amended recites a processor operable to generate data without generating an address of a destination of the data, to retrieve the generated data and to load the retrieved data into a buffer under the control of a first data-transfer object, to unload the data from the buffer under the control of a second data-transfer object, and to process the unloaded data without receiving the address of the destination of the data.

For example, referring, *e.g.*, to FIGS. 3-5 and paragraphs [62]-[63] and [82] of the patent application, in an embodiment, a processor 42 is operable to generate data under the control of a first thread 100_3 of an application 80 without generating an address of a second, destination thread 100_4 of the data. The processor 42 is further operable to retrieve the data and to load the data into a buffer 106_5 under the control of a first data-transfer object 86_{5a}, to unload the data from the buffer 106_5 under the control of a second data-transfer object 86_{5b}, and to process the unloaded data under the control of the destination thread 100_4 without receiving the address of the destination thread. An object factory 98 generates the data-transfer objects 86_{5a} and 86_{5b} and defines the buffer 106_5 from configuration data stored in the registry 72. Therefore, if one wants to change the destination of the data generated by the first thread 100_3, he need not modify, recompile, and retest the application 80; he needs only to modify the configuration data stored in the registry 72.

In contrast, Dretzka does not disclose a processor operable to generate data without generating an address of a destination of the data, to retrieve the generated data and to load the retrieved data into a buffer under the control of a first data-transfer object, to unload the data from the buffer under the control of a second data-transfer object, and to process the unloaded data without receiving the address of the destination of the data.

In the response to the final Office Action mailed November 17, 2009, Applicants' attorney argued that Dreztka discloses a buffer 120-4 (FIG. 5) and an input list 230-4 (FIG. 6) that store data packets each having a 3-byte packet header (FIG. 17) that includes an address that indicates the destination (*e.g.*, the logical channel LCN) for the data packet (*e.g.*, col. 7, lines 35-38 and lines 45-53, col. 8, line 55 – col. 9, line 12, and col. 9, line 60 – col. 10, line 11).

The Examiner disagreed with the Applicants' attorney's argument, however.

But as discussed in paragraphs [38], [62], [65], [66], and [67] of the previously submitted §1.132 Declaration of Mr. John Rapp, even assuming that Dretzka's 3-byte packet header does not include a data-destination address, Dretzka's processor 11 still generates data-destination addresses, and includes these data-destination addresses in the data packets, unlike the claim 10 processor, which does not generate data-destination addresses.

Dretzka discusses in detail only data transfers between two switching modules (*e.g.*, switching modules 10 and 20 per FIG. 2 of Dretzka). Dretzka's switching modules 10, 20, and 30 are compliant with the ISO OSI 7-layer model (*see*, *e.g.*, col. 1, lines 20-25 and the level designations in FIG. 4), and Dretzka discusses his layers 2-4 of this model in detail (*e.g.*, FIG. 4). The only functions that Dretzka attributes to his layers 2-4 in the transmitting module are functions for sending data packets over multiple physical links 40 to a receiving module (the layers 2-4 have additional functions to be compatible with the 7-layer model, but Dretzka is silent as to these functions). For example, referring to FIG. 2, the layers 2-4 of the module 10 operate to send data packets over links 40-0 – 40-4 to the module 20. And the only functions that Dretzka attributes to the layers 2-4 in the receiving module are functions for receiving the data packets over the multiple physical links 40 from the transmitting module. For example, referring to FIG. 2, the layers 2-4 of the module 20 receive data packets received over the links 40-0 – 40-4 from the module 10.

But referring to FIG. 1 of Dretzka, Dretzka's system includes at least three modules 10, 20, and 30.

Consequently, as discussed in paragraphs [38], [62], [65], [66], and [67] of the §1.132 Declaration, although Dretzka provides the details of only a data transfer between two modules, it is inherent that the processor 11 of Dretzka's transmitting module (*e.g.*, the module 10) must generate, for the data, an address header indicating the destination module (*e.g.*, the module 20 or the module 30) of the data, because without this address header, the layers 2-4 of the transmitting module would not "know" over which set of physical links (*e.g.*, the set 40-0 – 40-4 for the module 20 or the other links 40-5 – 40-m for the module 30) to send the data; and the processor 21 of the receiving module would have no way of confirming whether the receiving module is an intended recipient of the received data without receiving this address header.

In summary, even if Dretzka's 3-byte packet header (FIG. 17) is not an address for the data, it is at least inherent that Dretzka's processor 11 generates other address headers for the data packets, the address headers indicating to which module 20 or 30 the module 10 is sending the data, and that the processor 21 receives these address headers. Therefore, Dretzka at least fails to teach a processor operable to generate data without generating an address of a destination of the data, and to process the data without receiving the address of the destination of the data, as recited in claim 10.

Furthermore, Dretzka also fails to teach a processor operable to retrieve data and to load the retrieved data into a buffer under the control of a first data-transfer object, and to unload the data from the buffer under the control of a second data-transfer object.

The examiner seems to think that a data-transfer object can be any piece of software that loads data into a buffer or that unloads data from a buffer.

But as discussed below, the examiner is incorrect.

A data-transfer object is a software object, and a software object is a mechanism for binding data with methods that operate on that data (see, *e.g.*, Wikipedia and the reference "Design Patterns: Elements of Reusable Object-Oriented Software," selected passages of which are cited in the accompanying Information Disclosure Statement). Therefore, a software object is an independent mechanism that may be modified independently of other software with which the object interacts. For example, as discussed in paragraphs [62] – [63] and FIGS. 3-5 of the patent application, in an embodiment, an object factory 98 generates data-transfer objects 86_{5a} and 86_{5b} and defines the buffer 106_5 from configuration data stored in the registry 72. Therefore, the objects 86_{5a} and 86_{5b} and the buffer 106_5 may be independent of the software application 80. Consequently, if one wants to change the destination of the data generated by the first thread 100_3, he need not modify, recompile, and retest the application 80; he needs only to modify the configuration data stored in the registry 72.

In contrast, Dretzka fails to disclose that his system executes data-transfer objects for respectively loading and unloading a buffer. In fact, Dretzka discloses no details regarding the structure of the software executed by his system.

Claims 11-12, 14-15, and 17-18

These claims are patentable at least by virtue of their respective dependencies from claim 10.

Claim 37

Claim 37 as amended recites publishing data with an application that does not generate an address of a destination of the data, loading the published data into a first buffer with a first data-transfer object, and retrieving the published data from the buffer with a second data-transfer object.

In contrast, for reasons similar to those discussed above in support of the patentability of claim 10, Dretzka does not disclose publishing data with an application that does not generate an address of a destination of the data, and does not disclose loading and retrieving data from a buffer with respective data-transfer objects.

Furthermore, Dretzka does not disclose publishing data.

The examiner seems to be interpreting "publishing" data as "generating" data. But referring, *e.g.*, to "Design Patterns: Elements of Reusable Object-Oriented Software," selected portions of which are included with the accompanying IDS, "publishing" data implies that there is a subscriber to the data, and that the publisher and the subscriber are linked by the data-transfer objects recited in claim 37 such that the publisher need not generate the address of the subscriber.

In contrast, because Dretzka's modules 10, 20, and 30 are switching modules that do not "know" a priori the destination of the data that they receive for routing, these modules cannot publish data to a subscriber.

Claims 39-43

These claims are patentable by virtue of their respective dependencies from claim 37.

Claim 45

Claim 45 as amended recites receiving a message that includes data and that includes a message header that indicates a destination address of the data, loading into a buffer, with a first data-transfer object, the received data, unloading the data from the buffer with a second data-transfer object, and processing the unloaded data with a software application without the software application receiving the destination address of the data.

In contrast, for reasons similar to those discussed above in support of the patentability of claim 10, Dretzka does not disclose loading and retrieving data from a buffer with respective data-transfer objects, and does not disclose processing the unloaded data with a software application that does not receive the destination address of the data.

Furthermore, Dretzka does not disclose receiving a message.

The examiner seems to think that a message can be any grouping of data and a header.

But as discussed below, the examiner is incorrect.

A message is a complete data structure that is in a format compatible with the interface of a software object that receives or transfers the message (see, *e.g.*, Wikipedia and the reference "Design Patterns: Elements of Reusable Object-Oriented Software," selected passages of which are cited in the accompanying Information Disclosure Statement).

In contrast, Dretzka fails to disclose a complete data structure that is in a format compatible with a software-object interface. Dretzka teaches data groups that are purposely divided into packets for transmission over links 40; consequently, these divided data groups are not complete data structures. And Dretzka does not even discuss software objects or interfaces thereto, let alone a data structure that is compatible with a software-object interface.

<u>Claims 47-49</u>

These claims are patentable by virtue of their respective dependencies from claim 45.

<u>Rejection Of Claims 1-3 And 5-9 Under 35 U.S.C. § 103(a) As Being Obvious Over Dretzka In View Of U.S. 6,985,975 To Chamdani (Sections 42-51 of the Office Action)</u>

For the reasons below, the Applicants' attorney requests that the examiner withdraw these rejections.

<u>Claim 1</u>

Claim 1 as amended recites first and second parallel buffers respectively associated with first and second data-processing units, and a processor operable to publish data, to load at least a portion of the published data into the first buffer under the control of a first data-transfer object, to load at least the same portion of the published data into the second buffer under the control of a second data-transfer object, and to retrieve at least the portion of the published data from the first and second buffers under the control of third and fourth data-transfer objects, respectively.

For example, referring, *e.g.*, to FIGS. 3-5 and paragraphs [67] – [72] and [83] of the patent application, in an embodiment, a processor 42 is operable, under the control of an application thread 100_3, to publish data, and is operable, under the control of data-transfer object 86_{3a}, and to load at least a portion of the published data into a first buffer 106_3, which is associated with a first data-processing unit (*e.g.*, a first hardwired pipeline 74) within the pipeline accelerator 44. The processor is further operable, under the control of data-transfer object 86_{5a}, to load at least the same portion of the published data into a second buffer 106_5, which is associated with a second data-processing unit (*e.g.*, a second hardwired pipeline 74) within the pipeline accelerator and which is parallel to the first buffer 106_3, and is operable retrieve at least the portion of the published data from the first and second buffers 106_3 and 106_5 under the control of third and fourth data-transfer objects 86_{3b} and 86_{5b}, respectively.

In contrast, for reasons similar to those recited above in support of the patentability of claims 10 and 37, Dretzka does not teach or suggest loading data and retrieving data from buffers using data-transfer objects, and does not teach or suggest publishing data.

And neither does Chamdani teach or suggest loading data and retrieving data from buffers using data-transfer objects, or publishing data.

Furthermore, Dretzka does not include first and second parallel buffers respectively associated with first and second data-processing units, and a processor operable to load at least a portion of published data into the first buffer and to load at least the same portion of the published data into the second buffer as recited in claim 1. Referring to FIG. 5, even if the buffers, *e.g.*, 120-0 and 120-4, can be considered parallel and to be associated with different data-processing units, the processor 11 does not load the same data into these buffers.

And neither does Chamdani disclose first and second parallel buffers respectively associated with first and second data-processing units. In contrast, Chamdani's buffers (*e.g.*, FIFOs 102 and 103) are associated with a same data-processing unit (*e.g.*, whatever data-processing unit is connected to the single output of the coupler 110, and to FIG. 5, step 408).

In fact, Chamdani teaches away from first and second parallel buffers associated with different data-processing units. Chamdani is concerned with redundant data-transmission paths to a same destination to ensure reliable data transfer to that destination; but having parallel data paths (buffers) to different destinations would be devoid of redundancy, and thus would give rise to the same problem that Chamdani designed his system to solve.

Consequently, the combination of Dretzka and Chamdani does not render claim 1 obvious, and Chamdani actually teaches away from the buffer configuration recited in claim 1.

Claims 2-3 and 5-9

These claims are patentable at least by virtue of their dependencies from claim 1.

Rejection Of Claim 4 Under 35 U.S.C. § 103(a) As Being Obvious Over Dretzka In View Of Chamdani And Further In View Of The Examiner's Taking Of Official Notice (Sections 52-53 of the Office Action)

The Applicants' attorney maintains his objection to the Examiner's taking of official notice for this claim as discussed in at least the response to the Final Office Action mailed November 17, 2009. Furthermore, for the reasons below, the Applicants' attorney requests the examiner to withdraw this rejection.

Claim 4

Claim 4 is patentable by virtue of its dependency from claim 1.

Furthermore, referring to paragraph [17] of the previously submitted §1.132 Declaration, the application layer is layer 7.

And referring to, *e.g.*, paragraphs [32] and [33] of the §1.132 Declaration, Dretzka does not discuss layer 7.

Therefore, because Dretzka does not even discuss the application layer 7, there is nothing in Dretzka that would indicate it would be obvious to thread Dretzka's application.

Moreover, referring to paragraphs [27] – [28] of the §1.132 Declaration, it would not be obvious to thread any of the functions discussed in Dretzka because such threading would reduce the overall data-transmit/receive performance, by, for example, reducing the data-transmit/receive rate and increasing the required message-buffering space. That is, Dretzka at least inherently teaches away from threading the described data-transmit/receive functions because doing so would degrade, not improve, performance.

The examiner seems to argue that threading the data-transmit/receive tasks disclosed in Dretzka would reduce stall time in Dretzka's system.

But as Mr. Rapp explained during the September 27 interview, the examiner is incorrect.

The data-transmit/receive functions described by Dretzka are almost universally implemented as interrupt-service routines (ISRs), which typically run at a much higher priority than threads. For example, a network card sets a hardware interrupt when its transfer or receive buffer is full of data. In response to the interrupt, the processor suspends its other functions and services the interrupt by executing an ISR, which unloads the data from the buffer. Because all the data to be unloaded is available in the buffer, the ISR need not wait for any additional data; therefore, not only will the ISR not stall, it cannot stall. So threading such an ISR, which cannot stall, to prevent the ISR from stalling would add a completely unnecessary complexity to Dretzka's software, and would add a completely unnecessary delay to Dretzka's system; therefore, not only would one of skill in the art be unmotivated to thread Dretzka's ISR, he/she would recognize that threading Dretzka's ISR would be counterproductive and provide no advantage. Metaphorically speaking, it would be akin to fireproofing the interior lining of a swimming pool filled with water.

Consequently, at least because Dretzka inherently teaches away from threading any of her described functions, the combination of Dretzka, Chamdani, and the Examiner's official notice fails to render claim 4 obvious.

<u>Rejection Of Claims 13-14, 19-24, 38, 44, 46, 50, And 53-54 Under 35 U.S.C. § 103(a) As Being Obvious Over Dretzka In View Of The Examiner's Taking Of</u>

Official Notice (Sections 54-68 of the Office Action)

The Applicants' attorney objects to the examiner taking of official notice for these claims. Furthermore, for the reasons discussed below, the Applicants' attorney requests the examiner to withdraw these rejections.

Claim 13

Claim 13 is patentable at least by virtue of its dependency from claim 10, and for reasons similar to those recited above in support of the patentability of claim 4.

Claim 14

Claim 14 is patentable at least by virtue of its dependency from claim 10.

Claim 19

Claim 19 recites a processor operable to execute an application and first and second data-transfer objects, to publish data under the control of the application, to load the published data into a buffer under the control of the first data-transfer object, to retrieve the published data from the buffer under the control of the second data-transfer object, to construct a message under the control of the second data-transfer object, where the message includes the retrieved published data and information indicating a destination of the retrieved published data, and to drive the message onto a bus under the control of the communication object; claim 19 further recites a pipeline accelerator that includes the destination of data and that is operable to receive the message, to recover the data from the message, and to process the recovered data at the destination without executing a program instruction.

For example, referring, *e.g.,* to FIGS. 3-5 and paragraphs [67] – [72] of the patent application, in an embodiment, a processor 42 is operable to execute an application 80 and first and second data-transfer objects 86_{1a} and 86_{1b}, to publish data under the control of the application, to load the published data into a buffer 106_1 under the control of the first data-transfer object 86_{1a}, to retrieve the published data from the buffer under the control of the second data-transfer object, to construct a message that includes data and information indicating a destination of the data within a pipeline accelerator 44, and to drive the message onto a bus 50. The accelerator 44 is operable to receive the message from the bus 50, to recover the data from the message, and to process the recovered data at the destination without executing a program instruction.

In contrast, for reasons similar to those recited above in support of the patentability of claims 10 and 37, Dreztka does not disclose or suggest publishing data, and does not disclose or suggest any type of software object such as data-transfer or communications objects.

Furthermore, as discussed in paragraphs [86] – [95] of the §1.132 declaration and as discussed further below, Dretzka does not contain information sufficient to allow one of ordinary skill in the art to replace, with an FPGA, Dretzka's components that receive a message from a bus, that recover the received published data from the message, that provide the recovered data to a destination, and that process the recovered data at the destination, such that the FPGA can process the recovered data without executing a program instruction.

First, to so modify Dretzka to include an FPGA that could communicate via the messages disclosed in Dretzka would require constructing the FPGA interface according to the ISO OSI 7-layer model taught by Dretzka. But as discussed in paragraphs [86] – [95] of the §1.132 declaration, the teachings of Dretzka are insufficient for one of ordinary skill in the art to construct an FPGA interface according to the ISO OSI 7-layer model. Dretzka discusses only layers 2-4 of the 7-layer model, and for these layers only discusses some of the functions that they perform. Consequently, Dretzka provides none of the information that one of skill in the art would have needed to construct layers 1 and 5-7 of an FPGA interface, and at best provides only some of the information that one of skill in the art would have needed to construct layers 2-4 of an FPGA interface.

And second, as at least can be inferred from paragraphs [86] – [95] of the §1.132

declaration, Dretzka at least inherently teaches away from using an FPGA interface to communicate with a processor (*e.g.*, Dretzka's processor 11) via messages. Referring to Dretka's FIGS. 5-6, to be compliant with the ISO OSI 7-layer standard, the 7 layers in a transmitting module must be symmetrical to the 7 layers in the receiving module, and vice versa. That is, for example, if a layer in the transmitting module encrypts the data according to a particular algorithm, then the same layer in the receiving module must decrypt the data according to the same algorithm. Furthermore, still referring to FIGS. 5-6, to implement the 7 layers in the module 10 is relatively complex. But referring to FIGS. 1, 5, and 6, because all modules (*e.g.*, modules 10, 20, and 30) are the same, once the module design and debug is complete, it can be replicated to easily populate a system with as many modules as desired. Furthermore, the identical maintenance (*e.g.*, software updates) can be performed on all of the modules. But including in the system processor-based modules such as the modules 10, 20, and 30, and also including in the system FPGA-based modules, would require the design, debug, and maintenance of two module types for the system, which, in effect, approximately doubles the design effort as compared the system described in Dretzka. Furthermore, Dretzka gives no indication as to whether his system will benefit from replacing some of the processor-based modules (*e.g.*, 10, 20, and 30) with FPGA-based modules. Therefore, even if one of skill in the art could determine how to design an FPGA to communicate with Dretzka's processor-based switching modules 10, 20, and 30, because replacing some of the processor modules with FPGA modules would significantly increase the cost and effort to design, debug, and maintain Dretzka's system with no apparent offsetting benefit, Dretzka inherently teaches away from replacing any of its processor-based modules with FPGA-based modules.

In addition, the examiner's implication in section [57] of the office action that the §1.132 declaration effectively admits that claim 19 is obvious in view Dretzka is incorrect; Mr. Rapp's declaration provide facts in support of the claims being patentable over Dretzka.

Moreover, in section [57] of the office action, the examiner incorrectly equates network interface functionality with processing recovered data as recited in claim 19. The examiner states "it would have been obvious . . . at the time of the invention to modify Dretzka such that the network interface functionality is implemented on an FPGA, which in turn processes recovered data using programmed logic instead of instruction execution." That is, the examiner seems to state that if an FPGA implements a networking functionality without executing a program instruction, then it processes data without executing a program instruction. But the examiner is incorrect because he is ignoring the language of the claim. Claim 19 recites a pipeline accelerator that is operable to receive a message, to recover data from the message, and to process the recovered data without executing a program instruction. It appears that receiving the message and recovering the data from the message are what the examiner has labeled networking functions; therefore, because processing the recovered data is recited in addition to these functions, the examiner cannot read "to process the recovered data" on networking functions. And the Applicant's attorney cannot see how the examiner can take official notice that an FPGA is able to "process recovered data" without executing a programming instruction when the examiner's own FPGA prior art, U.S. 5,361,373 to Gilson, discloses an FPGA must execute program instructions to process data! *See, e.g.*, U.S. 5,361,373, FIG. 1, RISC processor 14 and Reconfigurable Instruction Execution Unit 16.

Furthermore, Dretzka does not disclose or suggest a processor operable to drive a message onto a bus under the control of a communication object.

In an embodiment, a communication object causes a processor to drive the message onto a bus according to a protocol (*e.g.*, PCI, PCI Express) that is compatible with the bus and the other components (*e.g.*, a pipeline-accelerator card) coupled to the bus. For example, as discussed in paragraphs [62] – [63] and FIGS. 3-5 of the patent application, an object factory 98 generates a communication object 88 from configuration data stored in a registry 72. Therefore, if one wants to change the bus protocol, he need not modify, recompile, and retest the application 80; he need only modify the configuration data stored in the registry 72. For example, one may wish to add to the system a pipeline-accelerator card that is compatible with a different bus protocol than the former pipeline-accelerator card. But he needs only to modify the configuration data stored in the registry 72 so that the object factory 98 will generate the communication object 88 to be operable according to the new bus protocol. In contrast, Dretzka would need to modify, re-debug, and recompile her application software if a new bus protocol were implemented.

Claims 20-21

These claims are patentable at least by by virtue of their dependencies from claim 19.

Claim 22

Claim 22 recites a pipeline accelerator operable to generate data without executing a program instruction, to generate a header including information indicating a destination of the data, to package the data and header into a message, and to drive the message onto a bus, and recites a processor operable to execute an application, first and second data-transfer objects, and a communication object, to receive the message from the bus under the control of the communication object, to load into a buffer, under the control of the first data-transfer object, the received data without the header, the buffer corresponding to the destination of the data, to unload the data from the buffer under the control of the second data-transfer object, and to process the unloaded data under the control of the application.

In contrast, for reasons similar to those recited above in support of the patentability of claims 10 and 19: (1) Dreztka does not disclose or suggest any type of software object such as data-transfer or communications objects; (2) Dretzka does not contain information sufficient to allow one of ordinary skill in the art to replace, with an FPGA, Dretzka's components that generate data without executing a program instruction, generate a header including information indicating a destination of the data, package the data and header into a message, and drive the message onto a bus; and (3) the Applicant's attorney cannot see how the examiner can take official notice that an FPGA is able to "generate data" without executing a programming instruction when the examiner's own FPGA prior art, U.S. 5,361,373 to Gilson, discloses an FPGA that must execute program instructions to generate data! *See, e.g.*, U.S. 5,361,373, FIG. 1, RISC processor 14 and Reconfigurable Instruction Execution Unit 16.

<u>Claims 23-24</u>

These claims are patentable at least by virtue of their dependencies from claim 22.

<u>Claim 38</u>

Claim 38 is patentable at least by virtue of its dependency from claim 37, and for reasons similar to those discussed above in support of the patentability of claim 4.

<u>Claim 44</u>

Claim 44 as amended is patentable at least by virtue of its dependency from claim 37.

<u>Claim 46</u>

Claim 46 is patentable at least by virtue of its dependency from claim 45, and for reasons similar to those discussed above in support of the patentability of claim 4.

<u>Claim 50</u>

Claim 50 is patentable at least by virtue of its dependency from claim 45.

<u>Claim 53</u>

Claim 53 recites generating with a pipeline accelerator, and without executing a program instruction, a message header that includes a destination of data, generating with the pipeline accelerator, and without executing a program instruction, a message that includes the header and the data, driving the message onto a bus with the pipeline accelerator, receiving the message from the bus with a communication object running on a processor, loading into a buffer with a first data-transfer object running on the processor the received data absent the header, the buffer being identified by the destination, unloading the data from the buffer with a second data-transfer object running on the processor, and processing the unloaded data with the processor.

In contrast, for reasons similar to those recited above in support of the patentability of claims 10 and 19: (1) Dreztka does not disclose or suggest any type of software object such as data-transfer or communications objects; (2) Dretzka does not contain information sufficient to allow one of ordinary skill in the art to replace, with an FPGA, Dretzka's components that generate a message header that includes a destination of data, and that generate a message that includes the header and the data, without executing a program instruction; and (3) the Applicant's attorney cannot see how the examiner can take official notice that an FPGA is able to "generate a message header" and "a message that includes the header" without executing a programming instruction because not only does the examiner's own FPGA prior art, U.S. 5,361,373 to Gilson, fail to disclose the generation of messages headers and messages, it also discloses an FPGA that must execute program instructions to generate data! *See, e.g.*, U.S. 5,361,373, FIG. 1, RISC processor 14 and Reconfigurable Instruction Execution Unit 16.

<u>Claim 54</u>

Claim 54 is patentable at least by virtue of its dependency from claim 53.

<u>Rejection Of Claims 51-52 Under 35 U.S.C. § 103(a) As Being Obvious Over Dretzka In View Of U.S. Patent 5,361,373 To Gilson (Sections 69-71 of the Office Action)</u>

<u>Claim 51</u>

Claim 51 recites publishing data with an application running on a processor, loading the published data into a buffer with a first data-transfer object running on the processor, retrieving the published data from the buffer with a second data-transfer object running on the processor, generating information that indicates a hardwired pipeline for processing the retrieved data, packaging the retrieved data and the information into a message, driving the message onto a bus with a communication object running on the processor, receiving the message from the bus, and processing the published data with the indicated hardwired pipeline without executing a program instruction, the indicated hardwired pipeline being part of a pipeline accelerator that includes a field-programmable gate array.

In contrast, neither Dreztka nor Gilson, viewed alone or in combination, renders claim 51 obvious.

For reasons similar to those recited above in support of the patentability of claims 10, 19, and 37: (1) Dretzka does not disclose or suggest publishing data; (2) Dreztka does not disclose or suggest any type of software object such as data-transfer or communications objects; and (3) Dretzka does not contain information sufficient to allow one of ordinary skill in the art to replace, with Gilson's FPGA, Dretzka's components that process data.

Furthermore, Dretzka does not disclose or suggest processing data without executing a program instruction.

And Gilson does not supply any of the teachings lacking in Dretzka. For example, although Gilson discloses an FPGA 12, this FPGA must execute instructions to process data as evidenced by the RISC processor 14 and Reconfigurable Instruction Execution Unit 16 of FIG. 1.

<u>Claim 52</u>

Claim 52 is patentable by virtue of its dependency from claim 51.

<u>New Claim 62</u>

Claim 62 is the same as claim 19 except that it recites "a <u>single</u> bus."

In contrast, Dretzka discloses multiple links 40, and its teachings that the examiner relies on to reject claim 19 would not be present if Dretzka's system included only one link 40.

<u>Section [74] Of The Office Action</u>

Mr. Rapp believes the examiner is incorrect, and restates that one of ordinary skill in the art would realize that it is inherent that OSI layers other than the layers 2-4 (these are the layers specifically disclosed in Dretzka) are inherently present in Dretzka's system, and that these other layers are relevant to the examination of the claims as discussed above, for example, in support of the patentability of claim 10.

<u>Section [75] Of The Office Action</u>

The Applicants' attorney has amended claims 10 and 37 to address the examiner's argument.

<u>Section [76] Of The Office Action</u>

The amendment to claim 45 renders the examiner's argument moot.

<u>Section [77] Of The Office Action</u>

The Applicants' attorney has amended claim 1 to replace "data processing unit" with “data-manipulation unit”. If the examiner believes that a “data processing unit” can read on a wire/bus, then he is requested to prove that a wire/bus can process data.

<u>Section [78] Of The Office Action</u>

The Applicants’ attorney has addressed the examiner’s argument above in support of the patentability of claim 4.

<u>Sections [79] – [81] Of The Office Action</u>

The Applicants’ attorney has addressed the examiner’s argument relative to buffers and data-transfer objects above in support of the patentability of claim 10.

CONCLUSION

In view of the foregoing, claims 2-9, 11-12, 14-15, 17-18, 20, 24, 38-44, 46-50, and 54 as previously pending, claims 1, 10, 13, 19, 21-23, 37, 45, and 51-53 as amended, and new claim 62 are in condition for allowance. Therefore, the issuance of a formal Notice of Allowance at an early date is respectfully requested. **If the Examiner does not agree that all claims are in condition for allowance, the Examiner is respectfully requested to telephone the undersigned prior to issuing an action rejecting the claims to schedule a telephone interview.**

In the event additional fees are due as a result of this amendment, you are hereby authorized to charge such payment to Deposit Account No. 07-1897.

DATED this 8th day of October 2010.

Respectfully submitted,

GRAYBEAL JACKSON LLP

/Bryan A. Santarelli/

Bryan A. Santarelli
Registration No. 37,560

Customer No. 00996

Graybeal Jackson LLP
400 - 108th Avenue NE, Suite 700
Bellevue, Washington 98004-5565
Telephone: 425.455.5575
Facsimile: 425.455.1046